UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/607,397	06/30/2000	Damon Barry	13768.132	9886

22913 7590 04/16/2004

WORKMAN NYDEGGER (F/K/A WORKMAN NYDEGGER & SEELEY)
60 EAST SOUTH TEMPLE
1000 EAGLE GATE TOWER
SALT LAKE CITY, UT 84111

EXAMINER	
KISS, ERIC B	

ART UNIT	PAPER NUMBER
2122	11

DATE MAILED: 04/16/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/607,397	BARRY ET AL.	
	Examiner	Art Unit	
	Eric B. Kiss	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *04 February 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4 and 7-27* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4 and 7-27* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *30 June 2000* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 4 February 2004 has been entered. Claims 1-4 and 7-27 are pending.

Response to Amendment

2. Applicant's amendments to claims 1 and 24 appropriately address the objections to claims 1 and 24, based on informalities, as detailed in the previous Office action. Accordingly, these objections are withdrawn in view of Applicant's amendments.

Response to Arguments

3. Applicant's arguments filed 4 February 2004 have been fully considered but they are not persuasive.

As previously asserted by the Examiner, the collection of APIs included with the TETware product provide the capability to build/link test case files into a format executable by the test case controller, regardless of the source language, e.g., C, C++, Shell, Korn Shell, or Perl, used to code the test cases (see, for example, section 2.4 of TET_UG describing the API components as linkable object code). The TETware product further has the capability to handle test cases that were not developed to conform to one of the included APIs (see, for example, section 2.4.4 of TET_PG describing the handling of non-API test cases). Regardless of which language is used to program a test case, the test case has to be built/linked by the test case controller prior to execution (see, for example, the description of build mode in section 6.2.3 of TET_UG). When a test case is built, it is linked with the appropriate test case manager module and API, if the test case is API-compliant, and transformed into an executable test case having an interface with which the test case controller, in execution mode, can interact. Note that the compiled test case no longer has any dependency on the source language used to create the source code for that test case (the test case controller, if directed, proceeds to execute the specified compiled test cases). Therefore, the Examiner asserts that the collection of APIs included with the TETware product, along with a build mode invocation of the test case controller, create an interface to a test case that is language and format independent.

Further, since the test suite of TETware is a directory hierarchy of test cases with a well-defined interface stored on a computer, the Examiner asserts that a TETware test suire is, in fact, a binary program module. Test cases with differing language are discussed above.

Claim Rejections - 35 USC § 102

4. The text of those sections of Title 35, U.S. Code not included in this action can be found in a prior Office action.

5. Claims 1, 2, 4, and 7-27 are rejected under 35 U.S.C. 102(b) as being anticipated by the TETware Release 3.3 software product (hereinafter TETware) released September 18, 1998 by The Open Group, as evidenced by: "TETware User Guide, Revision 1.2" (hereinafter TET_UG), "Release Notes for TETware Release 3.3" (hereinafter TET_RN), and "TETware Programmers Guide, Revision 1.2" (hereinafter TET_PG).

As per claim 1, TETware is disclosed with a computer system comprising:

a binary program module (test suite) storing a plurality of individually accessible test cases (see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed), each comprising a set of instructions for testing a feature of the computer program through a language and format independent interface, at least some of the individually accessible test cases differing from one another in language and format (the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG; different test cases inherently differ from one another in format, e.g., at a low level, they comprise different bit patterns, and at a higher level, they comprise different instructions or parameters; the use of different source languages to build cases is also

disclosed, e.g., C, C++, Shell, Korn Shell, or Perl; see, for example, section 2.4 of TET_UG describing the API components as linkable object code);

a harness, comprising a set of instructions that executes a test case hierarchy (test scenario) on the computer program using the corresponding language and format independent interface of each individually accessible test case in the test case hierarchy (test case controller; see sections 2.1 and 2.2 of TET_UG; the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG);

a connector, comprising a set of instructions that scans the plurality of test cases and extracts those test cases to be used to test the computer program to ensure that it processes as intended, the connector creating a hierarchy of test cases from those that are selected and extracted (see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed), and selectively integrates a generic interface between the one or more test cases and the harness regardless of the language or format in which the test cases were written (test case managers and API libraries; see section 2.4 of TET_UG; see also section 2.4.4 of TET_PG describing the handling of non-API test cases); and

a processor for executing the one or more test cases, the harness and the connector (inherent in the operation of the UNIX and WINDOWS operating systems used to implement TETware; see section 1.1 of TET_UG).

As per claim 2, TETware is further disclosed with the set of instructions of the harness and the set of instructions of the connector utilizing an architecture that defines a means for accessing a resource over a network (see section 2.6.3 of TET_UG).

As per claim 4, TETware is disclosed with a method comprising:

the connector scanning the binary program module (test suite) storing the plurality of individually accessible test cases, at least some of which differ from one another in language and format (the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG; different test cases inherently differ from one another in format, e.g., at a low level, they comprise different bit patterns, and at a higher level, they comprise different instructions or parameters; the use of different source languages to build cases is also disclosed, e.g., C, C++, Shell, Korn Shell, or Perl; see, for example, section 2.4 of TET_UG describing the API components as linkable object code), for one or more test cases of interest (see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed), each test case having a language and format independent interface for executing the test case on the computer program regardless of the language or format used to develop the test case (the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG);

the connector extracting the one or more test cases of interest from the binary program module (see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed);

the connector organizing one or more test cases into a test case hierarchy (test suite structure; see section 2.2 of TET_UG; see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed);

the connector interfacing a harness with the one or more test cases of interest (see section 6.4 of TET_UG; see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed), wherein the interfacing allows the harness to recognize and execute the one or more test cases of interest regardless of the language or format in which the one or more test cases of interest were developed (test case controller; see sections 2.1 and 2.2 of TET_UG; the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG); and

the harness traversing the test case hierarchy and executing each of the one or more test cases of interest to the computer program (see the description of the test case controller beginning on page 105 of TET_UG).

As per claim 7, TETware is further disclosed with a step of determining whether one or more of the test cases are identified as being deselected, wherein a deselected test case is not

executed on the computer program (see, for example, the "-n" command line option of the test case controller on page 107 of TET_UG).

As per claim 8, TETware is further disclosed with one or more test cases comprising a test suite in the hierarchy (see section 2.2. of TET_UG).

As per claims 9, TETware is further disclosed with one or more test suites comprising a test module in the hierarchy (test scenario; see section 2.2 of TET_UG).

As per claims 10 and 11, TETware is further disclosed with excluding test cases determined to be deselected from a selection of a test suite or scenario (see, for example, the "-n" command line option of the test case controller on page 107 of TET_UG).

As per claims 12-14, TETware is further disclosed with the step of traversing further including executing the one or more test cases on a thread pool comprising one or more threads, and further discloses testing single-threaded and multi-threaded (thread-safe) models (see section 17.4 of TET_PG).

As per claims 15-17, these are computer-readable medium versions of the method discussed above (claim 4), wherein all limitations have been addressed as set forth above. Furthermore, the use of such a computer-readable medium containing executable code is inherently necessary for the operation of the UNIX and WINDOWS operating systems used to implement TETware (see section 1.1 of TET_UG).

As per claims 18 and 19, see the disclosure applied above in the rejection of claims 8 and 9, respectively.

As per claim 20, TETware is further disclosed with user-selected test cases (see the description of the test case controller and command line usage beginning on page 107 of TET_UG).

As per claims 21-23, see the disclosure applied above in the rejection of claims 12-14.

As per claim 24, TETware is disclosed with a method comprising:

identifying one or more test cases from a binary program module (see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed) that stores a plurality of individually accessible test cases, at least some of which differ from one another in language and format (the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG; different test cases inherently differ from one another in format, e.g., at a low level, they comprise different bit patterns, and at a higher level, they comprise different instructions or parameters; the use of different source languages to build cases is also disclosed, e.g., C, C++, Shell, Korn Shell, or Perl; see, for example, section 2.4 of TET_UG describing the API components as linkable object code), each test case implementing a language and format independent interface for executing the test case on a computer program regardless of the language or format used to develop the test case (the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG);

translating the identified one or more test cases into a test case hierarchy (a test scenario see, for example, section 2.5.2 of TET_PG, which describes "Test scenario definitions" that specify which test cases of a test suite are to be executed);

interfacing the test case hierarchy in order to recognize and execute the one or more test cases regardless of the language or format in which the one or more test cases were written (test case controller; see sections 2.1, 2.2, and 2.4 of TET_UG; the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG); and

executing each of the one or more test cases in the test case hierarchy to test the computer program regardless of the language or format in which the one or more test cases were written (test case managers and API libraries; see section 2.4 of TET_UG; see also section 2.4.4 of TET_PG describing the handling of non-API test cases; the test cases are built and executed, regardless of their source language, through the same test case controller; see, for example, the description of build mode in section 6.2.3 of TET_UG).

As per claims 25-27, see the disclosure applied above in the rejection of claims 12-14.

Claim Rejections - 35 USC § 103

6. The text of those sections of Title 35, U.S. Code not included in this action can be found in a prior Office action.

7. Claim 3 is rejected under 35 U.S.C. 103(a) as being unpatentable over TETware and the associated cited documentation as applied to claim 1 above, and further in view of U.S. Patent No. 6,505,342 to Hartmann et al.

As per claim 3, TETware is disclosed with such a system (see disclosure applied above to claim 1), but is not expressly disclosed with a COM technology architecture. However, Hartmann et al. teach a system for testing components that use middleware, such as COM/DCOM (see column 2, line 61 through column 3, line 4). Therefore, it would have been obvious to one having ordinary skill in the computer art at the time the invention was made to modify the system of TETware to include a COM architecture as per the teaching of Hartmann et al. One would be motivate to do so to gain the advantage of supporting and testing implementations in a standardized object-oriented middleware.

Conclusion

8. Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Eric B. Kiss whose telephone number is (703) 305-7737. The Examiner can normally be reached on Tue. - Fri., 7:30 am - 5:00 pm. The Examiner can also be reached on alternate Mondays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Tuan Dam, can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is (703) 872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

EBK /EBK
April 9, 2004

Hoanguin Antony Nguyen Ba

ANTONY NGUYEN-BA
PRIMARY EXAMINER